Electrocardiògraf amb connectivitat Wi-Fi

Llorenç Fanals Batllori
Pol Fernández Rejón

Aplicacions industrials dels microprocessadors
UdG
14/01/2020

ÍNDEX

1. INTRODUCCIÓ

1.1. **Antecedents**

L'auge de les tecnologies i en especial la miniaturització i l'abaratiment de l'electrònica han fet que mesurar la freqüència cardíaca d'una persona sigui viable amb un equip modest, econòmicament parlant. No són poques les persones que actualment tenen algun tipus de polsera que els mesura les pulsacions per minut i els indica aquesta dada ja sigui amb una petita pantalla LCD o comunicant aquestes dades al mòbil.

Als hospitals ja fa molts anys que es mira el pols als pacients. Els equips existents permeten monitoritzar de forma eficaç el pols. La majoria d'aquests, però, manquen de connectivitat a Internet i no tenen la capacitat d'enregistrar un nombre considerable de dades.

1.2. **Objectiu**

L'objectiu d'aquest projecte és dissenyar i implementar un equip de monitorització de freqüència cardíaca amb connectivitat a Internet.

A nivell de hardware es busca aconseguir un dispositiu de dimensions reduïdes emprant components SMD i bateries d'alta densitat energètica. A nivell de software es programarà un microcontrolador que adquirirà les dades de pols i en farà el tractament corresponent. Un segon microcontrolador mostrarà en una pàgina web les dades ja tractades.

1.3. **Abast**

El maquinaria de la placa contindrà els reguladors de tensió necessaris per adequar la tensió de dues piles de 1,2 V a les tensions de treball dels diferents microcontroladors. Es disposarà d'una Arduino Nano per fer el processat de les dades, que vindran d'una petita placa anomenada AD8232. Un ESP8266 llegirà via I2C les dades de l'Arduino Nano, les guardàra en memòria i les mostrarà en una pàgina web.

En concret, es desitja veure dues gràfiques. Una que mostri les dades de freqüència cardíaca de l'últim minut aproximadament i una altra que mostri les dades de l'últim dia.

2. FUNCIONAMENT DEL COR HUMÀ I MONITORITZACIÓ

El cos humà està format per un nombre considerable d'òrgans, cadascun amb una funció específica. En l'actualitat hi ha un gran nombre de variables d'una persona que es poden mesurar, com poden ser la temperatura, l'oxigen en sang, la glucosa en sang, el pes, l'altura, el percentatge de greix corporal, la concentració de glòbuls vermells, la capacitat pulmonar...

Un òrgan vital de l'ésser humà és el cor i la magnitud que el caracteritza és la freqüència cardíaca, que normalment es dona en batecs per minut, o sigui, bpm.

2.1. **Electrocardiograma**

De forma teòrica, un electrocardiograma és una mesura indirecta de l'activitat elèctrica cardíaca. De fet, és la única mesura no invasiva de què es disposa per aquest fi. Permet identificar alteracions anatòmiques, el ritme, alteracions iòniques...

Durant la despolarització del miòcit cardíac es genera una diferència de potencial de 90 mV. El camp elèctric que es genera és captat pels elèctrodes.

Aquesta senyal elèctrica s'amplifica per tal d'aprofitar el rang dels convertidors analògics digitals de què disposen els electrocardiògrafs digitals.

El que s'espera veure en un electrocardiograma es mostra a la següent imatge:

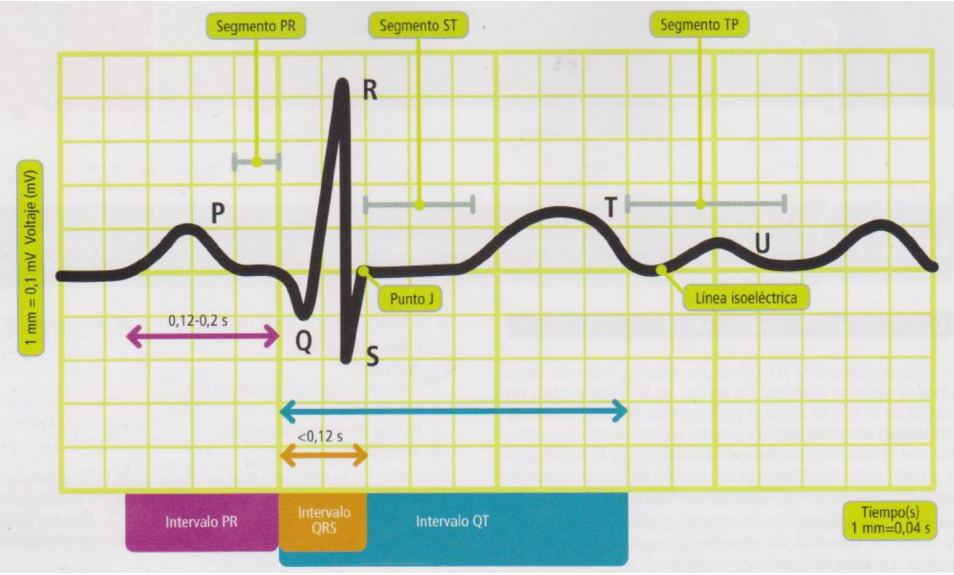

Figura 1. Senyal teòrica d'un ECG i els seus segments

Hi ha molta literatura que permet identificar malalties analitzant els diferents segments, la seva durada i l'amplitud de la senyal. Tot i que considerem que es podria programar un algorisme per fer un anàlisi de la senyal, ens hem centrat en calcular la freqüència cardíaca amb la fórmula de l'equació **??**.

$$\beta = \frac{1}{T} * 1000 * 60 \qquad \text{(Eq. 1)}$$

β: freqüència cardíaca (bpm).

T: període entre pic i pic (ms).

2.2. **Rang de freqüència cardíaca habitual**

Una persona al llarg de la seva vida sol tenir una freqüència cardíaca en repòs variable. La medicina no és una ciència exacte i per tant no és d'estranyar que cada autor o referència doni uns nivells habituals de freqüència cardíaca lleugerament diferents.

Per donar una referència, la majoria de la població té una freqüència cardíaca que es troba dins l'interval donat a la Taula **??**:

	Freqüència cardíaca (bpm)	
Edat	Mínim	Màxim
Recent nascuts, de 0 a 1 mes	70	190
Bebès de 1 a 11 mesos d'edat	80	160
Nens de 1 a 2 anys d'edat	80	130
Nens de 3 a 4 anys d'edat	80	120
Nens de 5 a 9 anys d'edat	75	115
Nens a partir de 10 anys i persones adultes	50	100
Esportistes amb bon entrenament cardiovascular	40	60

Taula 1. Freqüències cardíaques habituals segons l'edat

Com veiem, el rang de freqüència cardíaca és molt gran, especialment durant l'edat adulta. No es pot dir que una persona tingui una malaltia important per tenir el cor a 80 bpm, per exemple. Creiem més important conèixer l'evolució de la freqüència cardíaca d'una persona al llarg de la seva vida que no pas conèixer aquesta magnitud en un instant determinat.

És evident que tenir una freqüència cardíaca de 0 bpm o similar indica, sens dubte, que la persona ha patit una parada cardíaca.

Es diu que la freqüència cardíaca màxima d'una persona segons la seva edat ve regida

per l'equació **??**.

$$\beta_{opt} = 220 - e \qquad \text{(Eq. 2)}$$

e: edat de la persona (anys).

2.3. **Electrocardiògrafs actuals i solució proposada**

Tradicionalment un electrocardiograma es podia fer amb un electrocardiògraf analògic, el qual pot ser tant simple com un amplificador diferencial que dona la diferència de tensió entre dos elèctrodes i l'amplifica. La referència d'aquesta tensió pot ser el nivell de tensió d'un tercer elèctrode, el qual actua com a massa.

Aquesta configuració de 3 elèctrodes és la que nosaltres escollim, per bé que n'hi ha d'altres de viables que funcionen amb més elèctrodes i que poden ser més fiables.

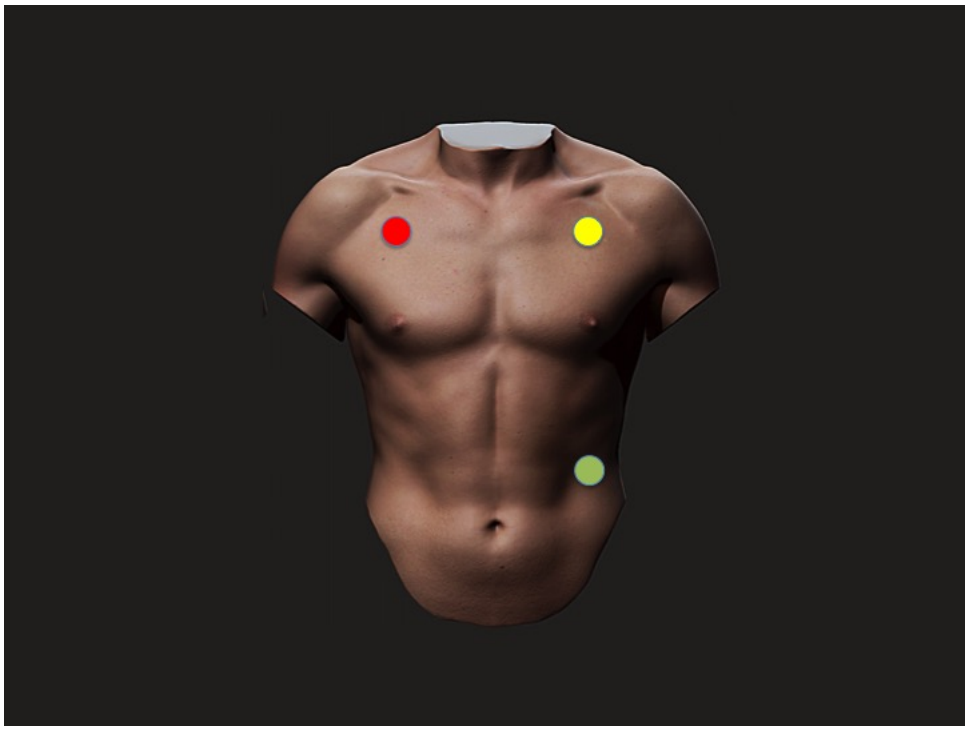

Figura 2. Col·locació de 3 elèctrodes

Un electrocardiògraf analògic no enregistra dades de freqüència cardíaca. Si es desplaça un tros de paper de forma constant i la senyal analògica de tensió s'amplifica adequadament i s'aconsegueix desplaçar de forma proporcional a la tensió llegida una agulla, la qual té un bolígraf o similar a un dels seus extrems amb què pot dibuixar un electrocardiograma.

Figura 3. Electrocardiògraf analògic

L'electrònica digital permet estalviar-se dibuixar la senyal de forma analògica sobre un paper. Es basa en mostrejar a alta velocitat la senyal analògica d'entrada i enregistrar les dades. D'aquesta manera es poden representar les dades captades en un pla, donant lloc a l'electrocardiograma.

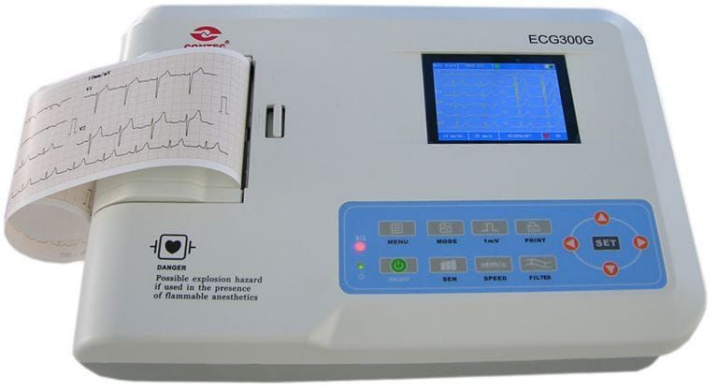

Figura 4. Electrocardiògraf actual

Els electrocardiògrafs digitals tenen l'avantatge de poder enregistrar dades, o sigui, tenen memòria. Els electrocardiògrafs analògics no ofereixen aquesta possibilitat. Tot i això, l'equip de la imatge, per exemple, tot i ser digital, funciona com un electrocardiògraf analògic en el sentit de què només imprimeix, no tracta ni emmagatzema de cap manera les dades.

Al llarg del treball es mostra com no només es dissenya un electrocardiògraf digital, sinó que aquest guarda dades que són consultables mitjançant una interfície clara, cosa que molts no electrocardiògrafs comercials no fan.

Tot i això, també donem l'opció a visualitzar l'electrocardiograma a través del por sèrie de l'Arduino Nano. Mostregem a una freqüència suficient, la qual assegura que no ens perdem cap pic ni cap dada significativa de l'electrocardiograma.

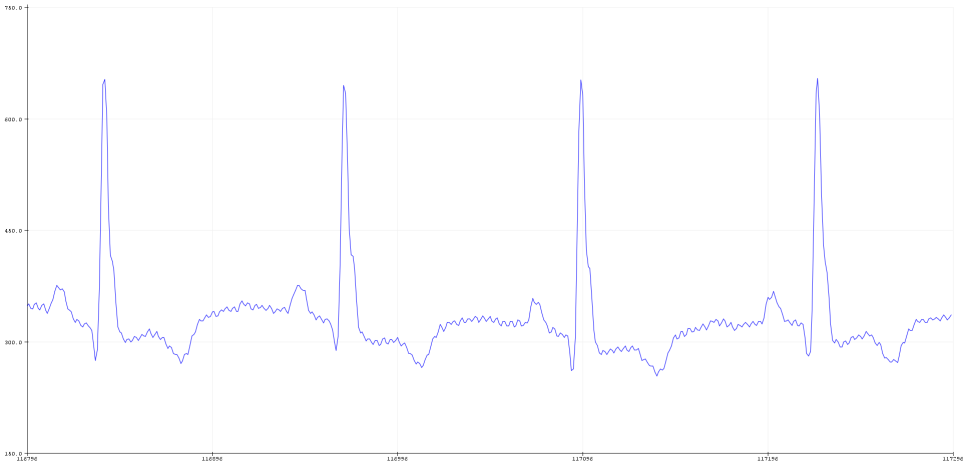

Figura 5. Electrocardiògraf donat per l'Arduino

3. PLACA ELECTRÒNICA D'ADQUISICIÓ DE DADES I COMUNICACIÓ

Tal com s'indica als objectius del projecte part del treball consisteix en desenvolupar una placa electrònica encarregada d'adquirir dades, en aquest cas de freqüència cardíaca.

Es vol que aquestes dades siguin accessibles i per això s'ha optat per dotar la placa d'un component de comunicació Wi-Fi gràcies al qual es podrà visualitzar una pàgina web amb les dades mesurades.

La placa disposa d'una part d'alimentació que s'encarrega d'adaptar les tensions als nivells correctes dels diversos components. Existeix comunicació I2C entre els dos microcontroladors. Un s'encarrega d'adquirir les dades i tractar-les, mentre que l'altre està dedicat a la comunicació Wi-Fi.

D'aquesta manera es poden realitzar tasques en paral·lel, cosa que seria impossible de fer amb un sol microcontrolador. Així, el microcontrolador que s'encarrega de la comunicació Wi-Fi pot esta donant servei a un client mentre el microcontrolador de l'Arduino Nano mostreja la senyal analògica.

Si s'intentés fer tot amb un sol microcontrolador, podria ser que durant la dècima de segon que es necessita per donar servei al client ens perdéssim un batec. La fiabilitat del sistema disminuïra, i no ho volem.

De forma esquemàtica es pot explicar el hardware mitjançant blocs.

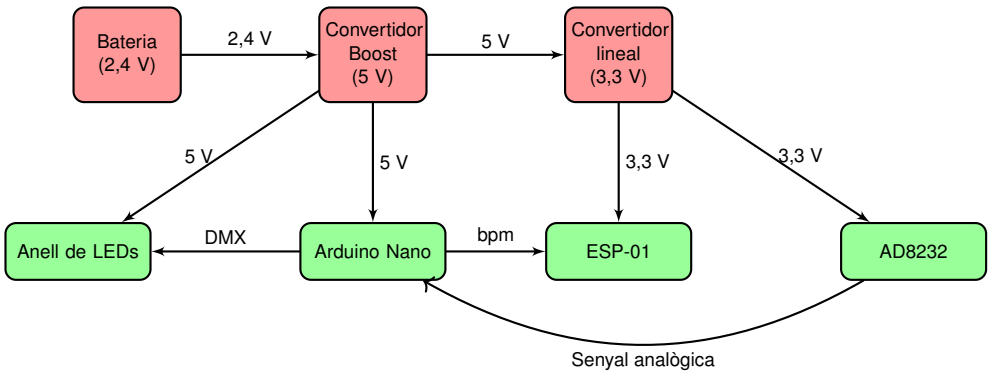

Figura 6. Esquema de blocs del hardware

3.1. **Alimentació**

L'etapa de potència del nostre equip s'encarrega de transformar la tensió d'aproximadament 2,4 V de dues piles en sèrie a 5 V mitjançant un convertidor Boost que admet una tensió d'entrada mínima de 1 V i pot donar fins a 500 mA a la seva sortida de 5 V. Té una eficiència elevada i la seva sortida és constant. Les dimensions són acceptables.

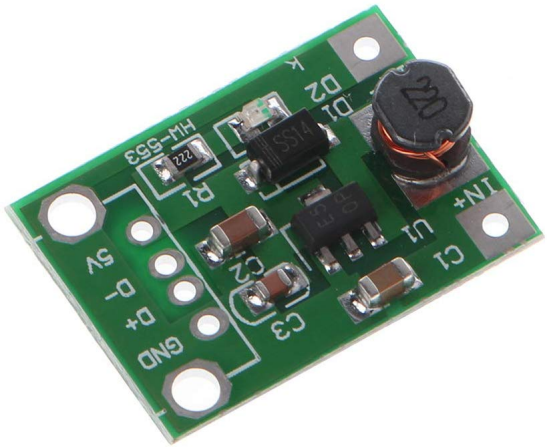

Figura 7. Regulador de tensió 5 V

Algunes característiques del regulador són les indicades a la Taula **??**.

Característica	Valor
Tensió de sortida	5 V
Corrent màxim de sortida	500 mA
Tensió mínima d'entrada	1 V
Precisió de regulació de voltatge	1%
Tensió màxima d'entrada	5 V
Topologia	Boost, font commutada

Taula 2. Característiques del regulador de tensió de la pila

No n'hi ha prou de tenir 5 V a la sortida d'aquest convertidor. Els 5 V són necessaris per alimentar l'Arduino Nano, ja que el seu microcontrolador ATMEGA328 funciona a 5 V. La petita placa que conté l'ESP8266 i la placa de l'ADS8232 van alimentades a 3,3 V. Per tant, és necessari disposar d'una tensió d'alimentació de 3,3 V.

El més directe seria fer servir el pin de 3,3 V de l'Arduino Nano, però comportaria un gran perill per aquesta placa de desenvolupament. Els 3,3 V de l'Arduino Nano venen d'un pin del mateix xip ATMEGA328. La màxima intensitat de sortida que pot donar és de 50 mA.

El component ESP8266 consumeix, en funcionament normal, uns 70 mA, així que no és una solució.

S'opta per un regulador lineal de bona eficiència anomenat AMS1117-3.3. Està especialment pensat per passar de tensions de 5 V a 3,3 V, que és exactament el que necessitem. A la seva entrada hi connectarem un condensador de 10 uF, així com a la seva sortida. Aquests condensadors evitaran la caiguda de tensió a la sortida en pics d'intensitat a la sortida.

S'ha escollit aquest component amb el package SMD per tal d'optimitzar al màxim l'espai.

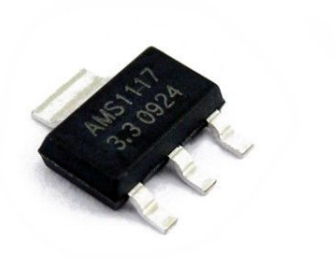

Figura 8. Regulador de tensió 3,3 V AMS1117 3,3

Algunes característiques del regulador són les indicades a la Taula **??**.

Característica	Valor
Tensió de sortida	3,3 V
Corrent màxim de sortida	800 mA
Tensió d'entrada	5 V
Precisió de regulació de voltatge	1,4%
Topologia	Lineal

Taula 3. Característiques del regulador de tensió de 5 V a 3,3 V

Noteu els 800 mA màxims de sortida. Si considerem que la placa amb l'ESP8266 pot arribar a, com a màxim, pics de 200 mA i que la placa d'instrumentació té consums habituals de 170 uA (si no es considera el LED que pot tenir en alguns models comercials), no hi ha dubte que aparentment no hi ha problemes.

En resum, podem mostrar una imatge que il·lustra de forma bastant intuïtiva les con-

nexions d'aquests equips d'alimentació. Permeten adaptar la tensió de la pila als diferents nivells que requereixen les diferents plaques.

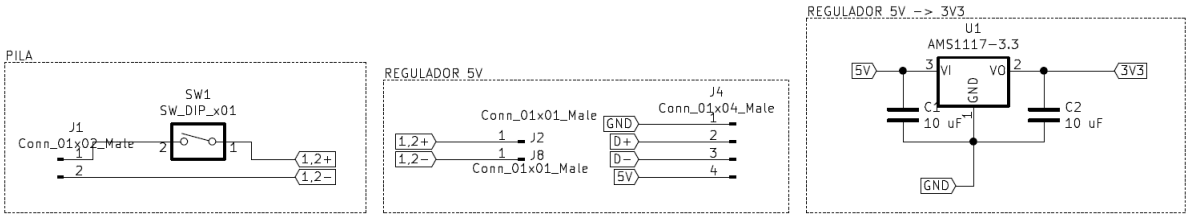

Figura 9. Etapes de potència de l'equip

3.2. **Instrumentació**

La part d'instrumentació de la placa dona una senyal analògica a partir de la senyal captada als tres elèctrodes. Per això es fa servir una placa de desenvolupament que conté un integrat anomenat AD8232.

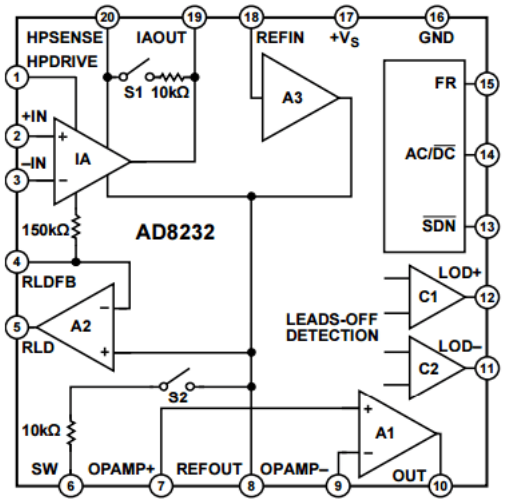

Figura 10. Diagrama de blocs de l'integrat AD82332

En essència, el que es fa és alimentar la placa i llegir les tensions als tres elèctrodes. Dues d'aquestes tensions es resten i el resultat s'amplifica amb un amplificador diferencial, la referència del qual és el tercer elèctrode.

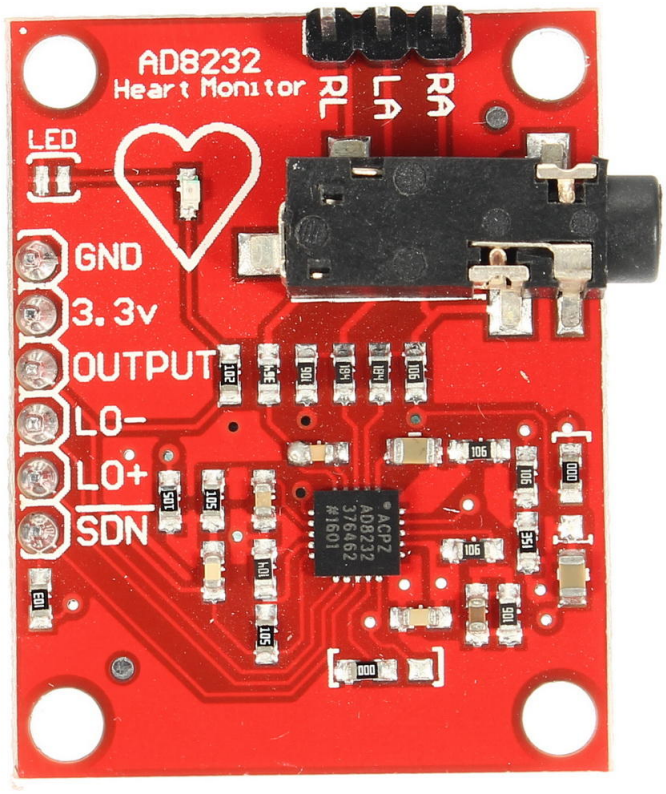

Figura 11. AD8232 en una placa de desenvolupament

La sortida analògica de la placa que mostrem és fruit de les 3 tensions, tal com acabem d'explicar. Si les limitacions d'espai no fossin un problema es podria haver afegit un filtre pas baix per eliminar la freqüència de 50 Hz, provinent de la xarxa elèctrica.

3.3. **Anell de LEDs**

S'ha previst disposar d'un anell de LEDs per visualitzar de forma ràpida i intuïtiva la freqüència cardíaca de la persona que porti el dispositiu. Per bé que no podrà conèixer amb excessiva precisió la freqüència cardíaca de la persona, se'n pot fer una bona idea.

L'anell que s'ha escollit és de la casa SparkFun. Té 16 LEDs 5050 disposats de forma circular. S'ha d'alimentar a 5 V i mitjançant un pin de DATA_IN es pot fer el control dels colors de tots els LEDs així com de la seva intensitat llumínica i la seva saturació.

Sparkfun té les seves llibreries com a codi obert, de forma que és molt fàcil testejar

exemples i entendre de forma pràctica com funcionen.

Figura 12. Anell de LEDs

La programació de l'anell s'ha fet de manera que cada LED té un color determinat. Si la freqüència és igual o més alta a la freqüència que simbolitza aquell LED, aquest s'encén. En cas contrari s'apaga. Cada vegada que hi ha un pols es fa com un refresc o "barrido" i s'encenen tants LEDs com cal. D'aquesta manera es dona una sensació de dinamisme.

El consum d'aquest anell és de 18 mA segons indica el fabricant. Hem pogut comprovar que aquest número s'ajusta molt al mesurat a la realitat, i ens va sorprendre que fos així, ja que de normal un LED d'aquest tipus consumeix cap a uns 20 mA en funcionament normal.

La raó que explica com és que el consum de 16 LEDs il·luminats a la vegada sigui tan baix és que cada LED s'il·lumina durant molt poc temps. Això es fa a alta freqüència i per tant l'ull humà ho interpreta com si cada LED sempre estigués encès.

3.4. **Comunicació**

L'ESP8266 permet establir comunicació Wi-Fi de forma efectiva. Després de fer algunes proves hem pogut determinar que el rang de comunicació possible de l'ESP8266 és proper al d'un telèfon mòbil convencional. La seva antena és una pista de circuit imprès. Actualment els mòbils també utilitzen aquest tipus d'antena.

L'integrat ESP8266 muntat en una placa amb antena, cristall oscil·lador, resistències de

pullup i condensadors de desacoblament es coneix com a ESP-01 i el seu pinout.

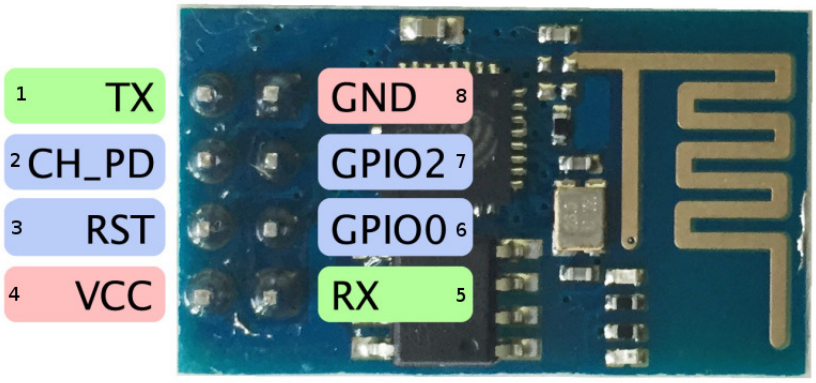

Figura 13. ESP-01

El fet de què aquesta placa només tingui 8 pins no vol dir que el component només tingui 8 pins, de fet en té al voltant de 30. La placa que utilitzem té dos pins d'alimentació, un pin de TX i un de RX, dos pins de caràcter general, un pin per fer reset i un pin per habilitar l'integrat.

El fabricant indica que es deixi una àrea lliure de coure sota l'antena per evitar interferències i problemes de comunicació.

Algunes característiques de l'ESP8266 venen donades a la Taula **??**.

Característica	Valor
Certificació	Wi-Fi Alliance
Freqüència de treball	2,4 GHz
Potència de l'emissor	17 dBm
Llindar de potència del receptor	-75 dBm
Rang de tensió d'alimentació	2,5 V a 3,6 V
Corrent en funcionament normal	80 mA

Taula 4. Característiques de l'integrat de comunicació ESP-8266

D'aquí la importància d'alimentar-lo amb 3,3 V enlloc dels 5 V que ens arriben a la sortida del primer regulador. Per això es fa servir el regulador de tensió que passa de 5 V a 3,3 V.

Un altre aspecte a destacar és la tensió dels pins digitals. En el nostre cas s'estableix comunicació I2C entre l'ESP-01 i l'Arduino Nano. Els pins digitals de l'Arduino Nano treballen a 5 V i els de l'ESP-01 en principi estan pensats per treballar a 3,3 V, que és la seva tensió d'alimentació. Així, podria haver-hi problemes si no es fes servir un adaptador de tensions.

El fabricant de l'ESP-01, però, ja va preveure que podria haver-hi conflicte i assegura que els pins digitals poden estar a 5 V sense problema de danyar l'integrat.

3.5. **Circuit imprès**

S'ha dissenyat un PCB a doble cara que conté les parts electròniques que s'han anat comentant.

Per bé que es podria haver ajuntat els esquemàtics de totes les plaques d'avaluació que fem servir en una sola placa de circuit imprès, això hauria portat un temps considerable per adquirir tots els components i testejar-los. A més, alguns components ens hauria sigut molt difícil, per no dir impossible, de soldar manualment.

Si aquest projecte seguís endavant i tingués el finançament necessari, llavors sens dubte s'optaria per contractar un servei integral en què ens fessin el PCB i ens hi soldessin els components.

Les pistes d'alimentació s'han fet més gruixudes que les de senyal digital. Les pistes de tensió de les plaques s'han fet una mica més gruixudes, tot i que la intensitat a través seu és petita. La separació d'una d'aquestes pistes amb la resta és generosa per evitar l'arc elèctric. A més, es preveu recobrir la placa amb una pel·lícula, cosa que disminuiria encara més el risc d'arc elèctric.

La majoria de components són SMD. Solen ser més barats i ocupen molt menys espai en comparació als components que travessen la placa. Així, les dimensions finals de la placa són de 4,5 x 4,1 cm aproximadament.

S'han seguit especificacions del fabricant pel disseny de la part de RF. Pel disseny de la part d'instrumentació s'ha tingut especial cura en utilitzar condensadors de desacoblament a les alimentacions dels integrats i situar-los prop d'aquests.

La vista superior de com quedaria la placa és la de la Figura **??**.

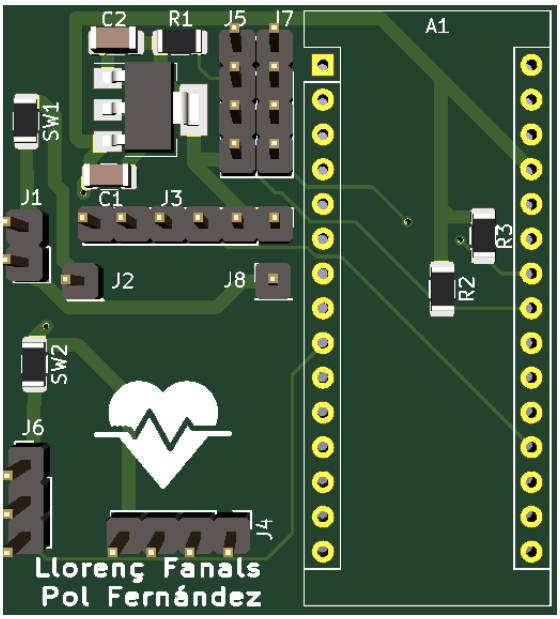

Figura 14. Vista 3D de la cara superior de la placa

No es disposa del model 3D de l'Arduino Nano.

Es pot observar com figuren una gran quantitat de connectors. La majoria simbolitzen allà on aniran connectades les plaques de desenvolupament. Algunes aniran amb pins de gran longitud per aconseguir tenir, a part d'aquesta placa "base", dos nivells més de plaques.

D'aquesta manera s'intenta optimitzar l'espai el màxim possible. Si es col·loquessin totes les plaques de desenvolupament de costat el resultat seria un equip amb molt poc gruix però molta superfície.

La serigrafia de què s'ha dotat la placa facilita les connexions de les diferents plaques de desenvolupament.

Pel que fa a la cara inferior, aquesta es mostra a la Figura **??**. Aquesta cara s'utilitza per fer passar alguna pista que d'altra manera congestionaria la cara superior. Per passar d'una capa a l'altra es fan servir via's. La resta de la cara superior conté el pla de massa i forats pels components through-hole.

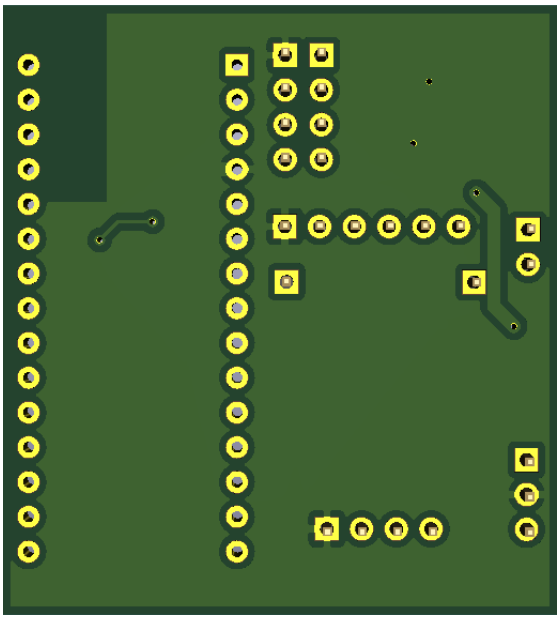

Figura 15. Vista de la cara inferior de la placa

Hem decidit encarregar aquesta placa a Xina. Per uns 20 € hem aconseguit 5 unitats i amb transport ràpid. El fabricant, JLCPCB, ha fabricat la placa en menys de 24 hores. L'enviament ha durat uns 3 dies. Considerem el resultat molt satisfactori:

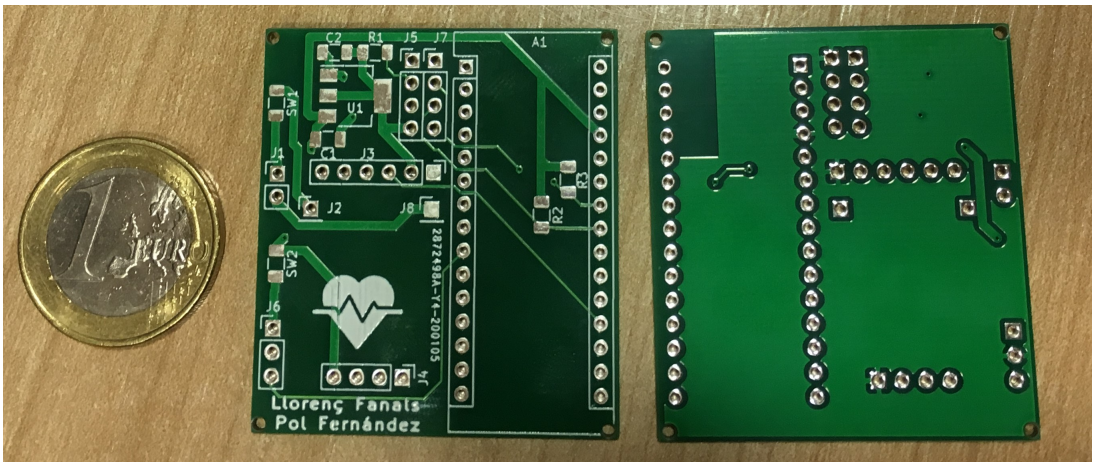

Figura 16. Vista de les dues cares del PCB

4. PROGRAMACIÓ

Tot el hardware explicat anteriorment ha d'anar recolzat del seu software. En aquest cas, el més primordial és mostrejar la senyal analògica per tal de poder determinar si aquesta supera un llindar de màxim o si disminueix d'un llindar de mínim. Això ho fa l'Arduino Nano, la qual disposa d'entrades analògiques.

En segon lloc cal que l'Arduino Nano controli els LEDs i n'encengui més o menys en funció de la freqüència cardíaca mesurada.

Mostrar la pàgina web se n'encarrega l'ESP-01, i hem comprovat que requereix d'uns 100 ms per fer-ho. La resta del temps es dedica a sol·licitar dades a l'Arduino Nano. Si aquesta no té cap nova dada de freqüència cardíaca per enviar, simplement no diu res.

4.1. Organigrama Arduino Nano

L'organigrama de la Figura **??** ens permet entendre a alt nivell el funcionament del programa que s'encarrega de llegir la tensió analògica provinent de l'ADS8232, calcular la freqüència cardíaca, controlar els LEDs i passar la freqüència cardíaca. A l'annex de programa figura el codi complet.

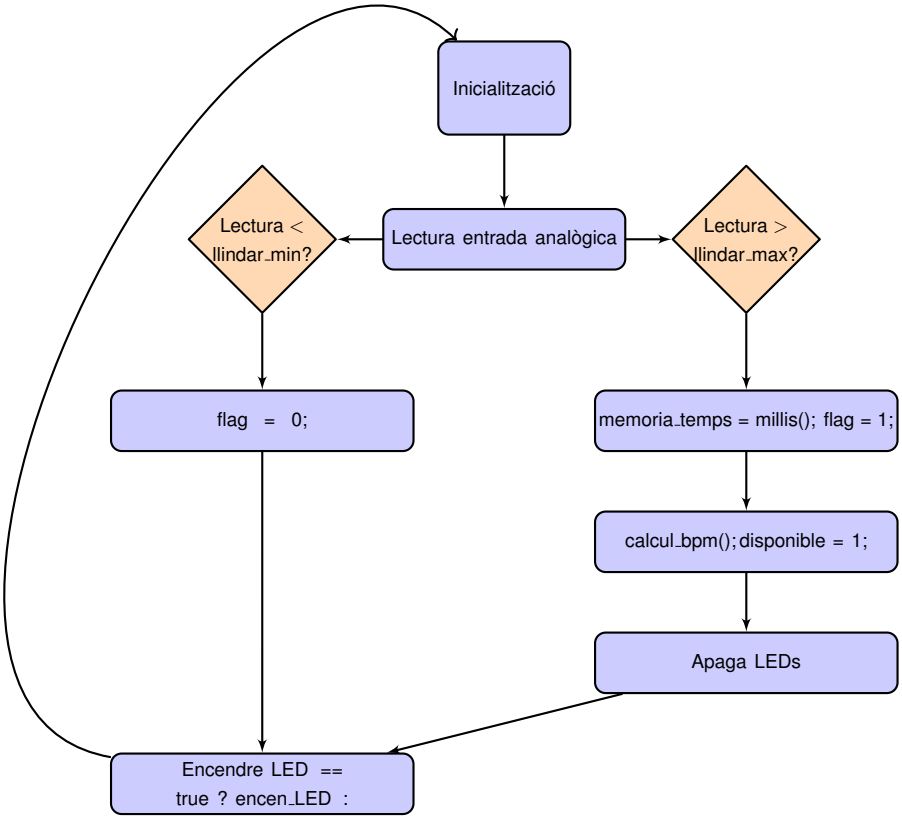

Figura 17. Organigrama del codi

Després de cada petició de la placa ESP-01 a l'Arduino, per I2C, s'executa una rutina d'interrupció que envia per I2C la freqüència cardíaca si encara no l'ha enviat. Si l'ha enviat no envia res.

4.2. **Programació Arduino Nano**

Tot el codi figura a l'annex. Creiem, però, que és important comentar aquells aspectes o fragments més importants.

Definim un color per cada LED de l'anell de LEDs i ho guardem en un vector.

```
int32_t c0 = strip.Color(255, 255, 0);
uint32_t c1 = strip.Color(178, 255, 0);
uint32_t c2 = strip.Color(55, 255, 0);
uint32_t c3 = strip.Color(6, 172, 38);
uint32_t c4 = strip.Color(0, 210, 255);
uint32_t c5 = strip.Color(8, 0, 255);
```

```
uint32_t c6 = strip.Color(1, 0, 255);
uint32_t c7 = strip.Color(107, 0, 222);
uint32_t c8 = strip.Color(101, 0, 144);
uint32_t c9 = strip.Color(255, 0, 208);
uint32_t c10 = strip.Color(255, 0, 199);
uint32_t c11 = strip.Color(123, 0, 96);
uint32_t c12 = strip.Color(128, 0, 46);
uint32_t c13 = strip.Color(223, 2, 10);
uint32_t c14 = strip.Color(223, 0, 0);
uint32_t c15 = strip.Color(255, 0, 0);
uint32_t leds[16]={c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10, c11, c12,
    c13, c14, c15};
```

Al **void setup()** definim l'adreça I2C que volem que tingui el dispositiu, en aquest cas hem escollit l'adreça 0x08.

```
void setup() {
  Wire.begin(8); //0x08=8, adreça que fem servir per comunicar amb l'ESP-01
// Wire.onReceive(receiveEvent);
  Wire.onRequest(sendEvent);
  ...
}
```

Al **void loop()** ens dediquem a llegir molt ràpidament la tensió analògica que ens arriba del sensor de freqüència cardíaca. Prèviament s'han definit dos llindars, un de mínim i un de màxim, que permeten identificar de forma correcta i robusta els pics, amb els quals es calcula la freqüència cardíaca en batecs per minut (bpm).

Per determinar si cal encendre LEDs o no es mira la freqüència cardíaca mesurada i el temps respecte l'última encesa de LEDs. Això ho fem per aconseguir un efecte d'encesa gradual, enlloc de veure com de cop s'encenen tots els LEDs.

```
  // Determinem si cal encendre leds o no
    if ((millis()-temps_memoria_2) > delay_entre_leds)
    {
      if (i<freq_bpm*16/130)
      {
       strip.setPixelColor(i, leds[i]);
       strip.show();
      i++;
      }
      temps_memoria_2 = millis();
    }
```

Per últim, definim una funció que s'encarrega de contestar amb la dada de freqüència cardíaca més recent, sempre que no l'hagi enviat prèviament.

```
// Envia la frequencia enlloc d'un numero random, i com a condicional té
    un boolea que diu si cal enviar o no
```

```
void sendEvent(int howmany){
  // Funció per respondre per I2C a l'ESP-01
  /*
  randNumber = random(40, 200); // Serial.print("generat: ");
     Serial.println(randNumber);

  cadena[2] = (randNumber - (randNumber/10)*10) + 48;
  cadena[1] = (randNumber/10 - (randNumber/100)*10) + 48;
  cadena[0] = (randNumber / 100) + 48;
  // Serial.println(cadena);

  if (randNumber >= 40){
    Wire.write(cadena); // Serial.println("major de 100");
  }
 // Wire.endTransmission(true);
 // Wire.write(cadena); // 3 bytes
 */

  cadena[2] = (freq_bpm - (freq_bpm/10)*10) + 48;
  cadena[1] = (freq_bpm/10 - (freq_bpm/100)*10) + 48;
  cadena[0] = (freq_bpm / 100) + 48;
  if (dada_enviada == 0){
    Wire.write(cadena);
    dada_enviada = 1;
  }

}
```

4.3. **Organigrama placa Wi-Fi**

L'organigrama de la Figura **??** és una forma simplificada d'explicar com s'ha programat l'ESP-01. A l'annex de programa figura el codi complet.

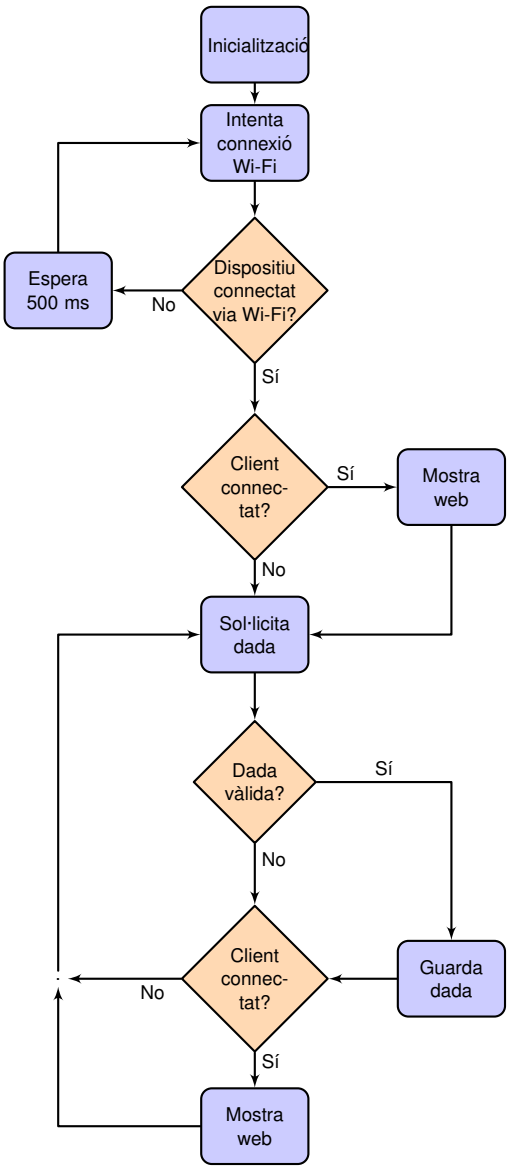

Figura 18. Organigrama del codi

4.4. **Programació placa Wi-Fi**

Quan s'inicia el programa el primer que es fa és inicialitzar variables i intentar connectar-se a la xarxa Wi-Fi de la qual s'ha definit la contrasenya i el nom, o més ben dit, l'SSID. Si el dispositiu és incapaç de connectar-se a la xarxa Wi-Fi ho segueix intentant cada mig segon.

```
// Ens connectem al Wi-Fi amb l'adreça i la contrasenya definits
Serial.print("Connectant a: ");
Serial.println(ssid); // Mostrem l'adreça del Wi-Fi
WiFi.begin(ssid, password); // Iniciem la comunicació

while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print("."); // Cada 0,5 s que passin sense connectar-se mostra
      un punt
```

```
}

// S'ha connectat
Serial.println("");
Serial.println("WiFi connectat");
Serial.println("Adreça IP: ");
Serial.println(WiFi.localIP());
server.begin();
```

Un cop té connectivitat Wi-Fi mira si té algun client connectat, i en cas de què sigui així li mostra la web.

Per defecte s'han emplenat els vectors de dades fictícies per tal de mostrar com es presenten les gràfiques que es fan amb JavaScript. A mesura que passin les hores aquestes dades s'aniran sobreescrivint per dades reals. La resta de la pàgina web està definida amb etiquetes HTML.

Per escriure les dades es recorren els vectors. Per una mateixa fila es miren totes les columnes, després es passa a la següent fila.

```
client.println("data.addRows([\n");
for (i = 0; i < files; i++) { // i = hora_actual ...
  client.println("[");
  client.println(String(vector2[i][0]));
  for (j = 1; j < 3; j++) {
    client.println(","); client.println(String(vector2[i][j]));
  }
  client.println("]"); client.println(","); client.println("\n");
}
```

Per tal de sol·licitar la freqüència cardíaca per I2C el que es fa és plantejar una funció anomenada **demana_dada()** que demana 3 bytes i si la dada és vàlida la guarda en memòria. A més, també s'encarrega de calcular la variància de les dades del primer vector, calcula la mitjana de freqüència cardíaca d'aquella hora i fa un calcula aproximat de la variància d'aquella hora. D'aquesta manera es preparen les dades per la pàgina web.

```
void demana_dada (){
  Wire.requestFrom(8, 3); //0x08 = 8;

  while (0 < Wire.available()) {
    char c = Wire.read();
    int nombre = c-48;
    // Serial.println(nombre);
    n_rebut = n_rebut*10 + (c-48);
  }

  if (n_rebut != -2523 && n_rebut != 0){
```

```
    vector[i_fila][0] = millis()/1000.0;
    vector[i_fila][1] = n_rebut;

       for (i=0; i<files; i++){
        mitjana = vector[i][1]*(1.0/(i+1.0)) + mitjana*i/(i+1.0);
       }
       int sumatori = 0;
       for (i=0; i<files; i++){
        sumatori += (mitjana - vector[i][1]) * (mitjana - vector[i][1]);
       }
       desvest = sumatori / (files-1);

    vector[i_fila][2] = 1.0*pow(desvest, 0.5);
  // 2 ms per fer els càlculs de mitjana i desvest, acceptable

    i_fila++;
    if (i_fila > files) {
     i_fila = 0;
    }

    Serial.println(n_rebut);

 //----- dades hora, 2a gràfica -----

    n_dades_hora++;
    vector2[hora_actual][0] = hora_actual;
    vector2[hora_actual][1] = n_rebut*1.0/(n_dades_hora) +
       vector[hora_actual][1]*(n_dades_hora-1)/(n_dades_hora);
    vector2[hora_actual][2] = vector[i_fila-1][2]*1.0/(n_dades_hora) +
       vector2[hora_actual][2]*(n_dades_hora-1)/(n_dades_hora);
 }

 else {
    Serial.println("és -2523"); // l'Arduino Nano no té cap dada nova,
       captem aquest número
 }

  n_rebut = 0;
  Wire.endTransmission(); // Afegit, no fa cap mal
}
```

Cada poc més de 50 ms es crida aquesta funció. La freqüència amb què es crida aquesta funció és més que suficient per poder garantir que no ens perdem cap dada de freqüència cardíaca, ja que el període d'un cor en repòs és de l'ordre de 1 s. Si algú té el cor molt accelerat, per exemple 120 bpm, això són 500 ms de període.

4.5. **Pàgina web**

A continuació es poden observar algunes captures de la pàgina web creada. A cada figura es mostren dues gràfiques, cada una és d'una branca i mostra la tensió de les 5 plaques que té connectades.

Quan s'entra a la web el primer que es veu és el que detalla la Figura **??**.

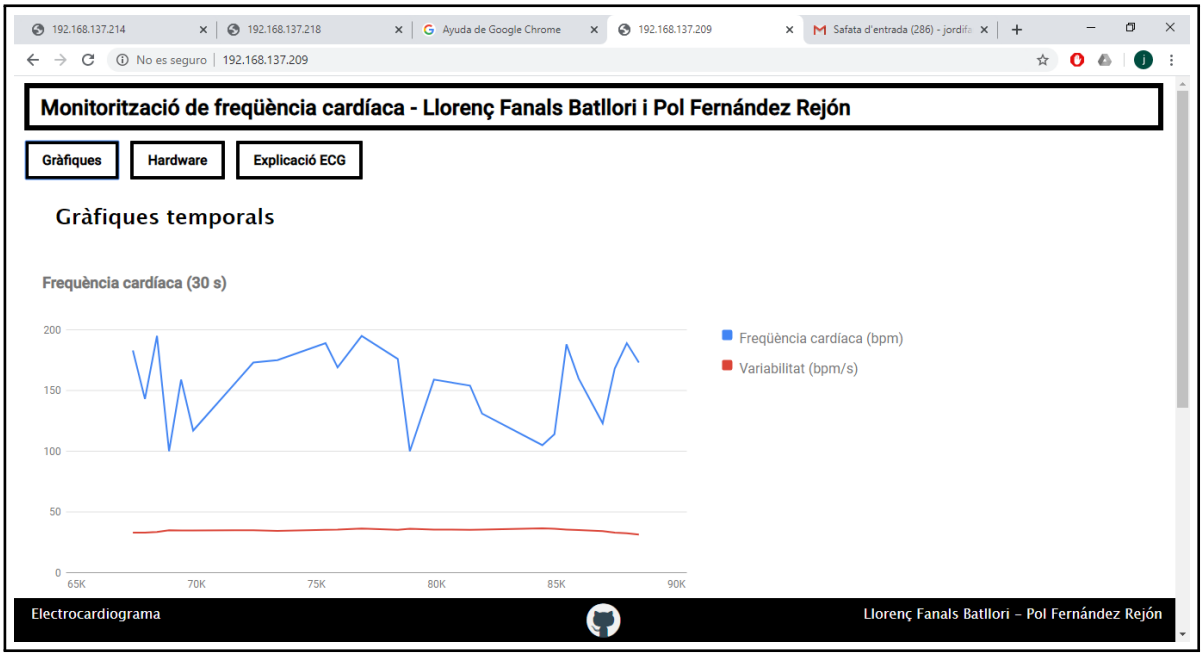

Figura 19. Gràfica de les dades més recents, web

El fet d'utilitzar JavaScript permet inserir contingut dinàmic. Al passar el ratolí per sobre una línia es ressalta la línia i els punts que la formen. És possible visualitzar el valor de cada punt col·locant el ratolí sobre seu A la Figura **??** s'aprecia.

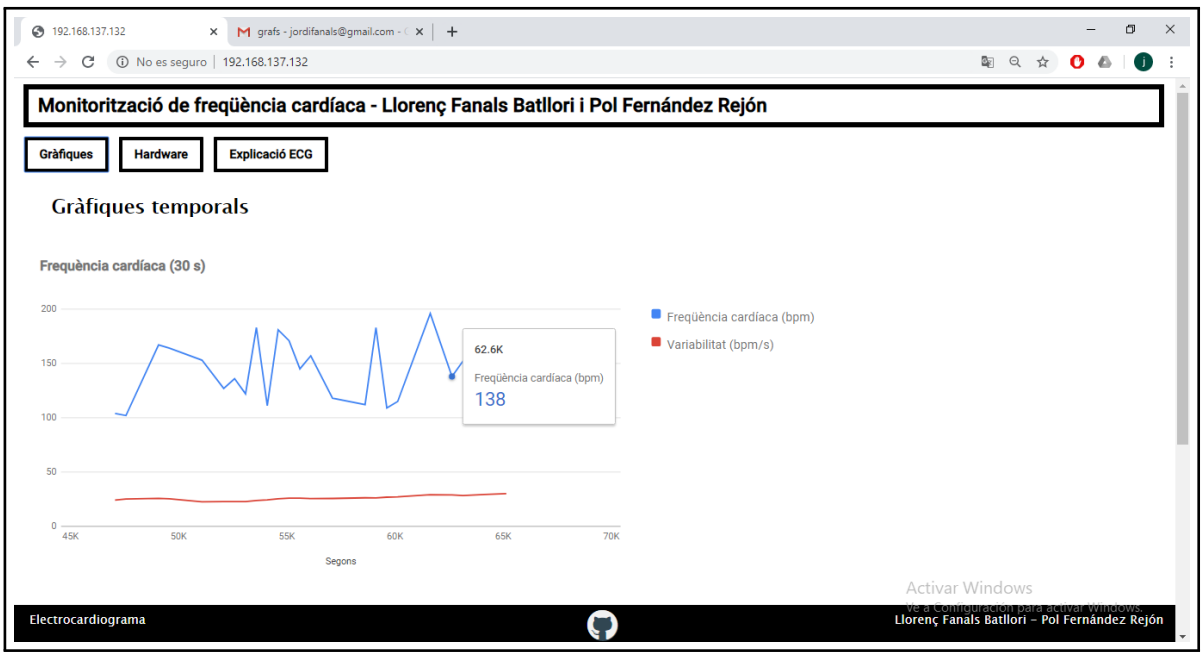

Figura 20. Gràfica ressaltant el valor d'un punt

Es pot navegar pels menús per veure un contingut o altre. Per exemple, podem demanar mirar l'apartat de hardware, i se'ns obra el següent:

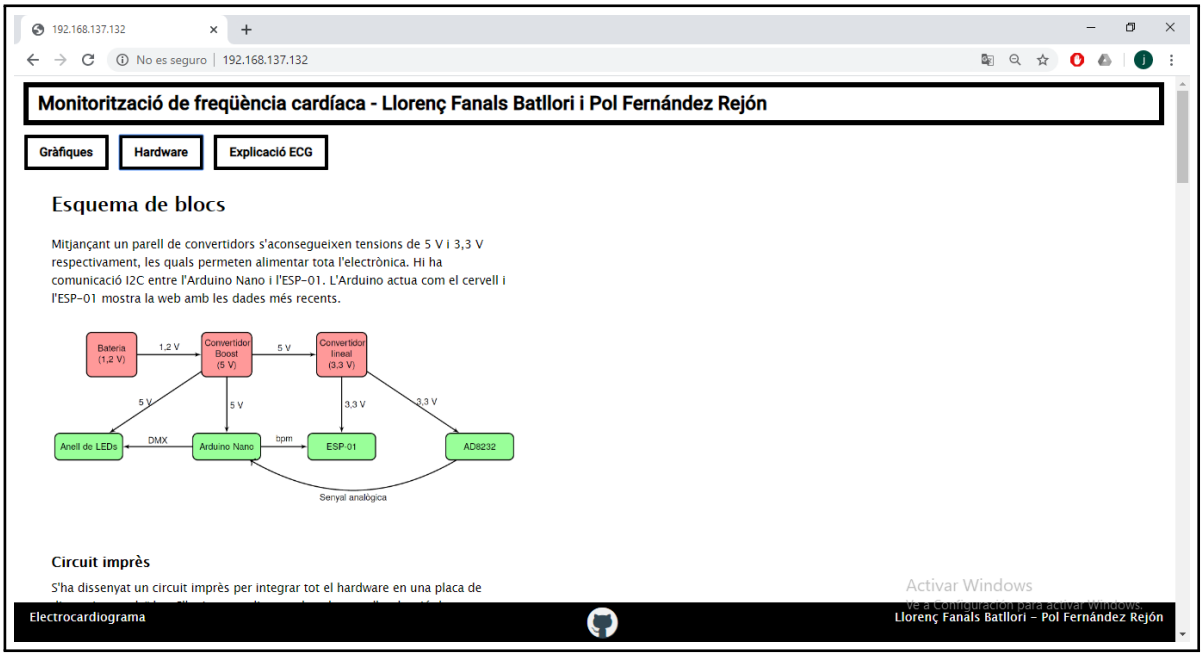

Figura 21. Apartat de hardware

5. CONCLUSIÓ

Per desenvolupar el projecte s'han consultat fonts fiables en el camp de la medicina. Tot i ser una disciplina de la qual tenim idees molt vagues, hem entès com són les diferències de tensió que es poden captar en diferents llocs del cos humà. Un cop parlem de tensions, parlem d'electrònica, camp en què sí que tenim coneixements.

El consum d'energia i les dimensions del hardware utilitzat han estat dues "constraints" que hem tingut en compte al llarg del desenvolupament. S'ha optat per una pila d'alta densitat energètica per tal d'optimitzar la relació energia/volum. Tot i utilitzar plaques de desenvolupament, s'ha aconseguit un volum de pocs centímetres cúbics, menor del de molts electrocardiògrafs comercials.

La programació dels microcontroladors s'ha fet per tal de fer en paral·lel les tasques d'adquirir dades i la de mostrar-les al client. D'aquesta manera augmenta la robustesa del sistema. S'ha programat una web en HTML que inclou estils definits en CSS i contingut dinàmic en JavaScript. Considerem que té una interfície atractiva i intuïtiva.

El client podrà conèixer amb precisió la freqüència cardíaca del pacient o persona que porti aquest aparell mitjançant una pàgina web. A la web apareixen les últimes dades de freqüència cardíaca així com les de l'últim dia. L'anell de LEDs és una forma molt visual de saber quina freqüència cardíaca té la persona.

Els autors considerem que s'han complert satisfactòriament els objectius marcats a l'inici del projecte.

Llorenç Fanals Batllori
Pol Fernández Rejón
Graduats en Enginyeria Electrònica Industrial i Automàtica

Girona, 14 de gener de 2020.

6. BIBLIOGRAFIA

GOOGLE. Line Chart by Google Charts. (https://developers.google.com/chart/interactive/docs/gallery/linechart?hl=es, 20 de novembre de 2019)

RANDOM NERD TUTORIALS. Build an ESP8266 Web Server - Code and Schematics. (https://randomnerdtutorials.com/esp8266-web-server/, 27 d'octubre de 2019)

SUÁREZ, A., et al. Manual AMIR ECG. Academia MIR. 2010.

VOWSTAR. NodeMCU Development Kit. (https://github.com/nodemcu/nodemcu-devkit-v1.0/blob/master/NODEMCU_DEVKIT_V1.0.PDF, 23 d'octubre de 2019)

7. GLOSSARI

API: Application Programming Interface.

CSS: Cascading Style Sheets DC: Corrent Continu.

ECG: Electrocardiogram

HTML: HyperText Markyp Language.

I2C: Inter-Integrated Circuit RF: Radiofreqüència.

SMD: Sourface Mount Device.

SSID: Service Set Identifier.

WIFI: Wireless Fidelity

A. PROGRAMES

A.1. Programa Arduino Nano

```
/* Llorenç Fanals Batllori
 * Pol Fernández Rejón
 * Electrocardiógraf amb connectivitat WiFi
 * Codi Arduino Nano
 */

#include <Wire.h> // Per comunicar per I2C
int randNumber; // Nombre aleatori que es generava per testejar
char cadena[3]; // Guarda la frequència cardíaca

#include <Adafruit_NeoPixel.h> // Llibreria per controlar l'anell de LEDs
#ifdef __AVR__
#include <avr/power.h>
#endif
#define LED_PIN 6       // S'utilitza el pin digital número 6
#define LED_COUNT 16    // Anell de 16 LEDs

Adafruit_NeoPixel strip(LED_COUNT, LED_PIN, NEO_GRB + NEO_KHZ800);

uint32_t off = strip.Color(0, 0, 0);
long int temps_memoria = 0;
long int temps_memoria_2 = 0;
int i = 0;
const int delay_entre_leds = 20;
const int delay_refresc = 1000;
int32_t c0 = strip.Color(255, 255, 0);
uint32_t c1 = strip.Color(178, 255, 0);
uint32_t c2 = strip.Color(55, 255, 0);
uint32_t c3 = strip.Color(6, 172, 38);
uint32_t c4 = strip.Color(0, 210, 255);
uint32_t c5 = strip.Color(8, 0, 255);
uint32_t c6 = strip.Color(1, 0, 255);
uint32_t c7 = strip.Color(107, 0, 222);
uint32_t c8 = strip.Color(101, 0, 144);
uint32_t c9 = strip.Color(255, 0, 208);
uint32_t c10 = strip.Color(255, 0, 199);
uint32_t c11 = strip.Color(123, 0, 96);
uint32_t c12 = strip.Color(128, 0, 46);
uint32_t c13 = strip.Color(223, 2, 10);
uint32_t c14 = strip.Color(223, 0, 0);
uint32_t c15 = strip.Color(255, 0, 0);
uint32_t leds[16]={c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10, c11, c12,
    c13, c14, c15};
long int freq_bpm;

long int temps_pic_anterior = 0;
int llindar_max = 600;
int llindar_min = 500;
int lectura_analogica = 0;
int diferencia_temps = 0;
bool pic = 0;

bool dada_enviada = 0;
```

```
void setup() {
  Wire.begin(8); //0x08=8, adreça que fem servir per comunicar amb l'ESP-01
// Wire.onReceive(receiveEvent);
  Wire.onRequest(sendEvent);
  Serial.begin(9600); // Habilitem el port sèrie, va bé per visualitzar
     l'electrocardiograma

  randomSeed(analogRead(1)); // Per generar nombres random

// -----------------------------

  #if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
    clock_prescale_set(clock_div_1);
  #endif

    strip.begin();     // Inizialitza l'anell
    strip.show();      // Apaga tots els LEDs
    strip.setBrightness(70); // Estableix la lluminositat, el màxim és 255,
       PWM

    // Illuminem cada LED amb el seu color
    strip.setPixelColor(0, c0);
    strip.setPixelColor(1, c1);
    strip.setPixelColor(2, c2);
    strip.setPixelColor(3, c3);
    strip.setPixelColor(4, c4);
    strip.setPixelColor(5, c5);
    strip.setPixelColor(6, c6);
    strip.setPixelColor(7, c7);
    strip.setPixelColor(8, c8);
    strip.setPixelColor(9, c9);
    strip.setPixelColor(10, c10);
    strip.setPixelColor(11, c11);
    strip.setPixelColor(12, c12);
    strip.setPixelColor(13, c13);
    strip.setPixelColor(14, c14);
    strip.setPixelColor(15, c15);

    strip.show(); // Indiquem que volem aplicar els canvis generats

    temps_memoria_2 = millis();

    temps_pic_anterior = millis();
}

void loop() {
 // sendEvent(2);

    lectura_analogica = analogRead(A0);
    Serial.println(analogRead(A0));

 // Analitzem si s'ha donat un pic
    if (lectura_analogica>llindar_max && pic == 0)
    {
      diferencia_temps = millis() - temps_pic_anterior;
      temps_pic_anterior = millis();
      pic = 1;
      freq_bpm = (60.0*1000.0 / diferencia_temps);
```

```
      dada_enviada = 0;
      i=0;
      strip.fill(off, 0, 16);
      strip.show();
    }
    else if (lectura_analogica < llindar_min && pic==1){
      pic = 0;
    }

  // Determinem si cal encendre leds o no
    if ((millis()-temps_memoria_2) > delay_entre_leds)
    {
      if (i<freq_bpm*16/130)
      {
       strip.setPixelColor(i, leds[i]);
       strip.show();
      i++;
      }
      temps_memoria_2 = millis();
    }

}

// TODO: enviar la frequencia enlloc d'un numero random, i com a
    condicional tenir un boolea que digui si cal enviar o no
void sendEvent(int howmany){
  // Funció per respondre per I2C a l'ESP-01
  /*
  randNumber = random(40, 200); // Serial.print("generat: ");
      Serial.println(randNumber);

  cadena[2] = (randNumber - (randNumber/10)*10) + 48;
  cadena[1] = (randNumber/10 - (randNumber/100)*10) + 48;
  cadena[0] = (randNumber / 100) + 48;
  // Serial.println(cadena);

  if (randNumber >= 40){
    Wire.write(cadena); // Serial.println("major de 100");
  }
 // Wire.endTransmission(true);
 // Wire.write(cadena); // 3 bytes
 */

  cadena[2] = (freq_bpm - (freq_bpm/10)*10) + 48;
  cadena[1] = (freq_bpm/10 - (freq_bpm/100)*10) + 48;
  cadena[0] = (freq_bpm / 100) + 48;
  if (dada_enviada == 0){
    Wire.write(cadena);
    dada_enviada = 1;
  }

}


/*
void receiveEvent(int howMany){
  String recibido;
  while (0 < Wire.available()) {
```

```
    char c = Wire.read();
    recibido += c;
  }
  Serial.print(recibido);
}
*/
```

A.2. **Programa ESP-01**

```
/*********
  Llorenç Fanals Batllori
  Pol Fernández Rejón
  Graduat en Enginyeria Electrònica Industrial i Automàtica
  20/11/2019
*********/
#include <Wire.h>
String recibido;

int i_fila = 0;
int i_columna = 0;

#include <ESP8266WiFi.h> // Es carrega la llibreria Wi-Fi

// Credencials de la xarxa Wi-Fi a què ens volem connectar
// const char* ssid = "DESKTOP-E5M4HBA 4049";
// const char* password = "E^1w1736";

const char* ssid = "DESKTOP-MQE758J 3309";
const char* password = "04)R936v";

// Port que volem utilitzar. El 80 és el port per defecte, així que
    teclejant la IP a un navegador en farem prou. Si fos un altre port la
    IP acabaria en ":número_port".
WiFiServer server(80);


unsigned long TempsActual = millis(); // Current time
unsigned long TempsAnterior = 0; // Previous time
const long TempsConnectat = 100; // Define timeout time in milliseconds
    (example: 2000ms = 2s)


#define files 24
#define columnes 5

float vector[files][columnes]; // vector de dades
float vector2[files][columnes]; // vector de dades
int i = 0; // iterador per files
int j = 0; // iterador per columnes

#define D0 16
#define D1 5
#define D2 4
#define D3 0

#define ENTRADA_ANALOGICA A0
```

```
unsigned int hores_posada_marxa = 10; // l'hora en què es fa la posada en
    marxa
unsigned int minuts_posada_marxa = 23; // a les 10:23 es fa la posada en
    marxa

unsigned int hora_actual;
float minuts_actual;
unsigned int millis_anteriors;

void inicialitza_vectors() { // Emplena els vectors de dades fictícies. A
    còpia d'hores s'aniran reemplaçant per dades reals
  //temps, bpm, pendent bpm
  vector[0][0] = 0; vector[0][1] = 0; vector[0][2] = 0;
  vector[1][0] = 0; vector[1][1] = 0; vector[1][2] = 0;
  vector[2][0] = 0; vector[2][1] = 0; vector[2][2] = 0;
  vector[3][0] = 0; vector[3][1] = 0; vector[3][2] = 0;
  vector[4][0] = 0; vector[4][1] = 0; vector[4][2] = 0;
  vector[5][0] = 0; vector[5][1] = 0; vector[5][2] = 0;
  vector[6][0] = 0; vector[6][1] = 0; vector[6][2] = 0;
  vector[7][0] = 0; vector[7][1] = 0; vector[7][2] = 0;
  vector[8][0] = 0; vector[8][1] = 0; vector[8][2] = 0;
  vector[9][0] = 0; vector[9][1] = 0; vector[9][2] = 0;
  vector[10][0] = 0; vector[10][1] = 0; vector[10][2] = 0;
  vector[11][0] = 0; vector[11][1] = 0; vector[11][2] = 0;
  vector[12][0] = 0; vector[12][1] = 0; vector[12][2] = 0;
  vector[13][0] = 0; vector[13][1] = 0; vector[13][2] = 0;
  vector[14][0] = 0; vector[14][1] = 0; vector[14][2] = 0;
  vector[15][0] = 0; vector[15][1] = 0; vector[15][2] = 0;
  vector[16][0] = 0; vector[16][1] = 0; vector[16][2] = 0;
  vector[17][0] = 0; vector[17][1] = 0; vector[17][2] = 0;
  vector[18][0] = 0; vector[18][1] = 0; vector[18][2] = 0;
  vector[19][0] = 0; vector[19][1] = 0; vector[19][2] = 0;
  vector[20][0] = 0; vector[20][1] = 0; vector[20][2] = 0;
  vector[21][0] = 0; vector[21][1] = 0; vector[21][2] = 0;
  vector[22][0] = 0; vector[22][1] = 0; vector[22][2] = 0;
  vector[23][0] = 0; vector[23][1] = 0; vector[23][2] = 0;

  vector2[0][0] = 0; vector2[0][1] = 0; vector2[0][2] = 0;
  vector2[1][0] = 1; vector2[1][1] = 0; vector2[1][2] = 0;
  vector2[2][0] = 2; vector2[2][1] = 0; vector2[2][2] = 0;
  vector2[3][0] = 3; vector2[3][1] = 0; vector2[3][2] = 0;
  vector2[4][0] = 4; vector2[4][1] = 0; vector2[4][2] = 0;
  vector2[5][0] = 5; vector2[5][1] = 0; vector2[5][2] = 0;
  vector2[6][0] = 6; vector2[6][1] = 0; vector2[6][2] = 0;
  vector2[7][0] = 7; vector2[7][1] = 0; vector2[7][2] = 0;
  vector2[8][0] = 8; vector2[8][1] = 0; vector2[8][2] = 0;
  vector2[9][0] = 9; vector2[9][1] = 0; vector2[9][2] = 0;
  vector2[10][0] = 10; vector2[10][1] = 0; vector2[10][2] = 0;
  vector2[11][0] = 11; vector2[11][1] = 0; vector2[11][2] = 0;
  vector2[12][0] = 12; vector2[12][1] = 0; vector2[12][2] = 0;
  vector2[13][0] = 13; vector2[13][1] = 0; vector2[13][2] = 0;
  vector2[14][0] = 14; vector2[14][1] = 0; vector2[14][2] = 0;
  vector2[15][0] = 15; vector2[15][1] = 0; vector2[15][2] = 0;
  vector2[16][0] = 16; vector2[16][1] = 0; vector2[16][2] = 0;
  vector2[17][0] = 17; vector2[17][1] = 0; vector2[17][2] = 0;
  vector2[18][0] = 18; vector2[18][1] = 0; vector2[18][2] = 0;
  vector2[19][0] = 19; vector2[19][1] = 0; vector2[19][2] = 0;
```

```
  vector2[20][0] = 20; vector2[20][1] = 0; vector2[20][2] = 0;
  vector2[21][0] = 21; vector2[21][1] = 0; vector2[21][2] = 0;
  vector2[22][0] = 22; vector2[22][1] = 0; vector2[22][2] = 0;
  vector2[23][0] = 23; vector2[23][1] = 0; vector2[23][2] = 0;
}

int n_rebut = 0;
float mitjana = 0;
float desvest = 0;
int n_dades_hora = 0;
float sumatori_2 = 0;

void setup() {
  Wire.begin(0, 2); // nodemcu: 4,5; esp-01: 0,2

  hora_actual = 0; // hores_posada_marxa
  minuts_actual = 0; // minuts_posada_marxa

  // Dades temporals dels vectors. Serveixen per mostrar com queden
     representades les gràfiques. S'aniran borrant les dades més antigues.
  inicialitza_vectors();

  Serial.begin(115200); // Habilitem el port sèrie a 115200 de baud rate

  // Ens connectem al Wi-Fi amb l'adreça i la contrasenya definits
  Serial.print("Connectant a: ");
  Serial.println(ssid); // Mostrem l'adreça del Wi-Fi
  WiFi.begin(ssid, password); // Iniciem la comunicació

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print("."); // Cada 0,5 s que passin sense connectar-se mostra
        un punt
  }

  // S'ha connectat
  Serial.println("");
  Serial.println("WiFi connectat");
  Serial.println("Adreça IP: ");
  Serial.println(WiFi.localIP());
  server.begin();

}

void demana_dada (){
  Wire.requestFrom(8, 3); //0x08 = 8;

  while (0 < Wire.available()) {
    char c = Wire.read();
    int nombre = c-48;
    // Serial.println(nombre);
    n_rebut = n_rebut*10 + (c-48);
  }

  if (n_rebut != -2523 && n_rebut != 0){

     vector[i_fila][0] = millis()/1000.0;
     vector[i_fila][1] = n_rebut;
```

```
        for (i=0; i<files; i++){
          mitjana = vector[i][1]*(1.0/(i+1.0)) + mitjana*i/(i+1.0);
        }
        int sumatori = 0;
        for (i=0; i<files; i++){
          sumatori += (mitjana - vector[i][1]) * (mitjana - vector[i][1]);
        }
        desvest = sumatori / (files-1);

      vector[i_fila][2] = 1.0*pow(desvest, 0.5);
    // 2 ms per fer els càlculs de mitjana i desvest, acceptable

      i_fila++;
      if (i_fila > files) {
        i_fila = 0;
      }

      Serial.println(n_rebut);

  //----- dades hora, 2a gràfica -----

      n_dades_hora++;
      vector2[hora_actual][0] = hora_actual;
      vector2[hora_actual][1] = n_rebut*1.0/(n_dades_hora) +
          vector[hora_actual][1]*(n_dades_hora-1)/(n_dades_hora);
      vector2[hora_actual][2] = vector[i_fila-1][2]*1.0/(n_dades_hora) +
          vector2[hora_actual][2]*(n_dades_hora-1)/(n_dades_hora);
  }

  else {
      Serial.println("és -2523"); // l'Arduino Nano no té cap dada nova,
          captem aquest número
  }

    n_rebut = 0;
    Wire.endTransmission(); // Afegit, no fa cap mal
}

void loop() {
  // Wire.beginTransmission(8);//0x08 = 8;
  // Wire.write("esp to uno \n");
  // Wire.endTransmission();
  // int temps_xyz = millis();

  demana_dada();
  delay(50); // 100

//
    ------------------------------------------------------------------------------

  WiFiClient client = server.available(); // Escolta si hi ha clients

  if (client) {                    // Si es connecta un nou client,
    Serial.println("Nou client.");
    String LiniaActual = "";       // una cadena memoritza la informació
        enviada pel client
    TempsActual = millis();
```

```
TempsAnterior = TempsActual;
while (client.connected() && TempsActual - TempsAnterior <=
    TempsConnectat) { // Si estem connectats i no han passat els
    milisegons que indica TempsConnectat,

  TempsActual = millis();
  if (client.available()) {    // Si el client ens passa informació,
    char c = client.read();    // llegim un caràcters ascii (un byte)
    Serial.write(c);           // i el mostrem per pantalla
    if (c == '\n') {           // Si rebem un canvi de línia com a
        caràcter,
      // és el final de la petició HTTP
      if (LiniaActual.length() == 0) {
        // Ara responem donant un OK i indicant el content type, volem
            una pàgina html. Finalment una línia en blanc, és el protocol
        client.println("HTTP/1.1 200 OK");
        client.println("Content-type:text/html");
        client.println("Tancant connexió");
        client.println();
        client.println("<!DOCTYPE html><html>");
        client.println("<head><meta name=\"viewport\"
            content=\"width=device-width, initial-scale=1\">");
        client.println(" <meta charset=\"UTF-8\">\n<meta
            http-equiv=\"Content-type\" content=\"text/html;
            charset=UTF-8\">");
        client.println("<link rel=\"icon\" href=\"data:,\">");
        client.println(" <link rel=\"stylesheet\"
            href=\"https://use.fontawesome.com/releases/v5.7.2/css/all.css\"
            integrity=\"sha384-fnmOCqbTlWIlj8LyTjo7mOUStjsKC4pOpQbqyi7RrhN7udi
        9RwhKkMHpvLbHG9Sr\" crossorigin=\"anonymous\">\n""");
        client.println(" <script type=\"text/javascript\"
            src=\"https://www.gstatic.com/charts/loader.js\"></script>\n
            <script type=\"text/javascript\">\n
            google.charts.load('current', {'packages':['line']});\n
            google.charts.setOnLoadCallback(drawChart);\n\n");

        // Definim la gràfica de la primera branca
        client.println(" function drawChart() {\n\n var data = new
            google.visualization.DataTable();\n data.addColumn('number',
            'Segons');\n data.addColumn('number', 'Frequència cardíaca
            (bpm)');\n data.addColumn('number', 'Variabilitat (bpm/s)');");

        client.println("\n data.addRows([\n");
        for (i = i_fila; i < files; i++) {
          client.println("[");
          client.println(String(vector[i][0]));
          for (j = 1; j < 3; j++) {
            client.println(","); client.println(String(vector[i][j]));
          }
          client.println("]"); client.println(","); client.println("\n");
        }

        for (i = 0; i < i_fila; i++) {
          client.println("[");
          client.println(String(vector[i][0]));
          for (j = 1; j < 3; j++) {
            client.println(","); client.println(String(vector[i][j]));
          }
```

```
          client.println("]"); client.println(","); client.println("\n");
        }

      client.println("]);\n\n\n var options = {\n 'width': 1000,\n
         'height': 400,\n  chart: {\n     title: 'Frequència cardíaca (30
         s)',\n bold: true, \n // subtitle: 'in millions of dollars
         (USD)'\n width: 100,\n },\n  titleTextStyle: {\n   bold: true,\n
            fontSize: 18,\n  }\n   //  width: 900,\n // height: 500\n
         };\n\n   var chart = new
         google.charts.Line(document.getElementById('linechart_material'));\n\n
         chart.draw(data, google.charts.Line.convertOptions(options));\n
         }\n </script>");
      // Definim la gràfica de la segona branca
      client.println(" \n\n <script type=\"text/javascript\"
         src=\"https://www.gstatic.com/charts/loader.js\"></script>\n
         <script type=\"text/javascript\">\n
         google.charts.load('current', {'packages':['line']});\n
         google.charts.setOnLoadCallback(drawChart);\n \n\n function
         drawChart() {\n\n var data = new
         google.visualization.DataTable();\n data.addColumn('number',
         'Hora');\n data.addColumn('number', 'Freqüència cardíaca
         (bpm)');\n data.addColumn('number', 'Variabilitat (bpm/s)');");
/*
        client.println("data.addRows([\n");
        for (i = hora_actual+1; i < files; i++) { // i = hora_actual ...
          client.println("[");
          client.println(String(vector2[i][0]));
          for (j = 1; j < 3; j++) {
            client.println(","); client.println(String(vector2[i][j]));
          }
          client.println("]"); client.println(","); client.println("\n");
        }
*/
        client.println("data.addRows([\n");
        for (i = 0; i < files; i++) { // i = hora_actual ...
          client.println("[");
          client.println(String(vector2[i][0]));
          for (j = 1; j < 3; j++) {
            client.println(","); client.println(String(vector2[i][j]));
          }
          client.println("]"); client.println(","); client.println("\n");
        }
/*
        for (i = 0; i < hora_actual+1; i++) {
          client.println("[");
          client.println(String(vector2[i][0]));
          for (j = 1; j < 3; j++) {
            client.println(","); client.println(String(vector2[i][j]));
          }
          client.println("]"); client.println(","); client.println("\n");
        }
*/
   client.println(" ]); \n\n\n\n var options = {\n 'width': 1000,\n
      'height': 400,\n chart: {\n title: 'Freqüència cardíaca (1 dia)',\n
      // is3D: true\n // subtitle: 'in millions of dollars (USD)'\n },\n
      titleTextStyle: {\n bold: true,\n fontSize: 18,\n }\n // width:
      700,\n // height: 400\n };\n\n var chart = new
      google.charts.Line(document.getElementById('linechart_material2'));\n\n
```

```
    chart.draw(data, google.charts.Line.convertOptions(options));\n }\n
    </script>\n\n </head>\n\n <style>\n .content {\n max-width: 100%;\n
    margin: left;\n }\n </style>\n\n<!--
##############################################
################################### -->\n\n <style>\nul {\n
    list-style-type: none;\n margin: 0;\n padding: 0;\n overflow:
    hidden;\n background-color: #333;\n}\n\nli {\n float: left;\n}\n\nli
    a {\n display: block;\n color: white;\n text-align: center;\n
    padding: 14px 16px;\n text-decoration: none;\n}\n\n/* Change the
    link color to #111 (black) on hover */\nli a:hover {\n
    background-color: #f3f3f3;\n}\n\n.active {\n background-color:
    #4CAF50;\n }\n\n li {\n border-right: 1px solid #bbb;\n }\n \n
    li:last-child {\n border-right: none;\n }\n\n ul {\n position:
    fixed;\n top: 0;\n width: 100%;\n }\n/*\n div{\n display:
    none;\n}*/\n \n.button {\n align-items: right;\n font-size: 16px;\n
    display:inline-block;\n padding:0.55em 1.0em;\n border:0.25em solid
    #000000;\n margin:0.3em 0.3em 0.3em 0.3em;\n border-radius:0.05em;\n
    box-
sizing: border-box;\n text-decoration:none;\n
    font-family:'Roboto',sans-serif;\n font-weight:800;\n
    color:#000000;\n text-align:center;\n transition: all 0.2s;\n\n
    background-color: #ffffff;\n }\n .button:hover{\n color:#ffffff;\n
    background-color:#000000;\n }\n @media all and (max-width:30em){\n
    .button{\n display:block;\n margin:0.4em auto;\n }\n }\n\n p.solid
    {\n font-size: 25px;\n font-weight: bold;\n margin:0.3em 0.25em
    0.3em 0.3em;\n font-family:'Roboto',sans-serif;\n border:0.25em
    solid #000000;\n padding:0.25em 0.5em;\n margin-left: 3pt;\n
    }\n\n\n\n p.text_limitat {\n max-width: 600px;\n margin-left:
    30pt;\n font-family: \"Lucida Sans Unicode\", \"Lucida Grande\",
    sans-serif;\n font-size: 15px;\n }\n\n h2.text_limitat {\n
    max-width: 600px;\n margin-
left: 30pt;\n font-family: \"Lucida Sans Unicode\", \"Lucida Grande\",
    sans-serif;\n font-size: 25px;\n }\n\n img.text_limitat {\n
    max-width: 600px;\n margin-left: 30pt;\n font-family: \"Lucida Sans
    Unicode\", \"Lucida Grande\", sans-serif;\n font-size: 25px;\n
    padding-top: 10px;\n }\n\n h3.text_limitat {\n max-width: 600px;\n
    margin-left: 30pt;\n font-family: \"Lucida Sans Unicode\", \"Lucida
    Grande\", sans-serif;\n font-size: 18px;\n margin-bottom: -5pt;\n }
    h2.titol {\n max-width: 600px;\n margin-left: 15pt;\n font-family:
    \"Lucida Sans Unicode\", \"Lucida Grande\", sans-serif;\n font-size:
    25px;\n margin-bottom: -5pt;\n }\n \n </style>\n\n<style>\n .footer
    {\n position: fixed;\n left: 0;\n bottom: 0;\n width: 100%;\n
    background-color: black;\n color: white;\n text-align: center;\n
    height: 35px;\n padding: 10px;\n padding-top: 6px;\n font-family:
    \"Lucida Sans Unicode\", \"Lucida Grande\", sans-serif;\n
    }\n\n\n\n\n\n .s-m{\n margin: 0px auto;\n justify-content:
    space-around;\n display: flex;\n max-width: 80px;\n display:
    block;\nmargin: 0 auto;\n}\n.s-m a{\n text-decoration: none;\n
    font-size: 40px;\n color: #f1f1f1;\n width: 40px;\n height: 40px;\n
    text-align: center;\n transition: 0.4s all;\n line-height: 40px;\n
    cursor: pointer;\n background: #314652;\n border-radius:
    50%;\n}\n.s-m a:hover{\n transform: scale(1.25);\n}\n\n\n\n
    </style>\n\n\n\n<!-- ##############################################
#####################################
  -->\n\n\n <body class=\"content\">\n\n <p class=\"solid\">\n
      Monitorització de freqüència cardíaca - Lloren&ccedil Fanals
      Batllori i Pol Fernández Rejón\n </p>\n \n \n <button
      class=\"button\"
```

```
    onclick=\"MostraGrafiques()\">Gràfiques</button>\n <button
    class=\"button\" onclick=\"MostraImatge()\">Hardware</button>\n
    <button class=\"button\" onclick=\"Explicacio()\">Explicació
    ECG</button>\n\n <script>\n function MostraGrafiques() {\n var x =
    document.getElementById(\"hardware\");\n x.style.display =
    \"none\";\n x = document.getElementById(\"teoria\");\n
    x.style.display = \"none\";\n\n x =
    document.getElementById(\"titol_grafs\");\n x.style.display =
    \"block\";\n x =
    document.getElementById(\"linechart_material2\");\n
    x.style.display = \"block\";\n x =
    document.getElementById(\"linechart_material\");\n x.style.display
    = \"block\";\n }\n </script>\n\n <script>\n function
    MostraImatge() {\n var x =
    document.getElementById(\"hardware\");\n x.style.display =
    \"block\";\n x = document.getElementById(\"teoria\");\n
    x.style.display = \"none\";\n\n x =
    document.getElementById(\"titol_grafs\");\n x.style.display =
    \"none\";\n x =
    document.getElementById(\"linechart_material2\");\n
    x.style.display = \"none\";\n x =
    document.getElementById(\"linechart_material\");\n x.style.display
    = \"none\";\n }\n </script>\n\n <script>\n function Explicacio()
    {\n var x = document.getElementById(\"hardware\");\n
    x.style.display = \"none\";\n x =
document.getElementById(\"teoria\");\n x.style.display =
    \"block\";\n\n x = document.getElementById(\"titol_grafs\");\n
    x.style.display = \"none\";\n x =
    document.getElementById(\"linechart_material2\");\n
    x.style.display = \"none\";\n x =
    document.getElementById(\"linechart_material\");\n x.style.display
    = \"none\";\n }\n </script>\n\n <!-- <img
    src=\"https://drive.google.com/uc?export=view&id=
1XgS6bALyKzA9_3eu245chrkyhIlQpjpq\" style=\"width: 50%;\"
    alt=\"Flowers in Chania\"> --> \n <!-- <h2
    align=\"margin-left\">Pol Fernández Rejón</h2> --> \n <div
    id=\"titol_grafs\"><h2 class=\"text_limitat\">Gràfiques
    temporals</h2> </div>\n\n <div id=\"linechart_material\"
    style=\"width: 800px; height: 400px; padding: 25px\"></div> \n
    <div id=\"linechart_material2\" style=\"width: 800px; height:
    400px; padding: 25px\"></div> \n\n <div id=\"hardware\"> \n <h2
    class=\"text_limitat\">Esquema de blocs</h2>\n <p
    class=\"text_limitat\">Mitjançant un parell de convertidors
    s'aconsegueixen tensions\n de 5 V i 3,3 V respectivament, les
    quals permeten alimentar tota l'electrònica.\n Hi ha comunicació
    I2C entre l'Arduino Nano i l'ESP-01. L'Arduino actua com el
    cervell\n i l'ESP-01 mostra la web amb les dades més recents.
    </p>\n <img class=\"text_limitat\"\n
    src=\"https://drive.google.com/uc?export=view&id=
1OwWSyZsfkwHrD8OPpXs3E0eJzi3jnsBT\"\n alt=\"Flowers in Chania\">\n\n
    <p class=\"text_limitat\"> </br></p> <!-- <img
    src=\"https://drive.google.com/uc?export=view&id=15-EkLWMhYaRsv-dtbyrlKOrbD7d
    style=\"width:
500px; height: 500px; margin-left: 100px; padding: 25px\"
    alt=\"Flowers in Chania\"> --> \n <h3
    class=\"text_limitat\">Circuit imprès</h3>\n <p
    class=\"text_limitat\">S'ha dissenyat un circuit imprès per
    integrar tot el hardware en una placa\n de dimensions reduïdes.
```

S'ha intentat disposar les plaques d'avaluació de manera que \n s'optimitzi l'espai el màxim possible. La placa és a dues cares. \n </p>\n <img class=\"text_limitat\"\n src=\"https://drive.google.com/uc?export=view&id=
1zm5T1PXHEUKQYz4tofg1PaqOihKiulTZ\"\n alt=\"Flowers in Chania\">\n\n <!-- BATERIA-->\n\n <h3 class=\"text_limitat\">Bateria 1,2 V</h3>\n <p class=\"text_limitat\">La bateria de 1,2 V és pila de 2800 mAh de capacitat. Ofereix\n una molt bona densitat d'energia, ja que té les dimensions estàndard d'una pila AA.\n És recarregable i permet alimentar el nostre equip durant més d'un dia de forma continuada,\n fins i tot si es tenen els LEDs encesos durant 24 hores. \n </p>\n <img class=\"text_limitat\"\n src=\"https://drive.google.com/uc?export=view&id=
1su7yWmxdUWt62ViUL51esPzXSx__obBr\"\n alt=\"Flowers in Chania\">\n <p class=\"text_limitat\">Passades 24 hores es recomana que es recarregui la pila per evitar \n que deixi de funcionar l'electrònica. No és desitjable perdre dades emmagatzemades a la memòria\n RAM de l'ESP-01.\n </p>\n\n <!-- CONVERTIDOR 5 V-->\n <h3 class=\"text_limitat\">Convertidor Boost</h3>\n <p class=\"text_limitat\">L'etapa de potència del nostre equip s'encarrega de transformar\n la tensió d'aproximadament 1,2 V de la pila a
5 V mitjançant un convertidor Boost que\n admet una tensió d'entrada mínima de 1 V i pot donar fins a 500 mA a la seva \n sortida de 5 V. Té una eficiència elevada i la seva sortida és constant. Les dimensions\n són acceptables.\n </p>\n <img class=\"text_limitat\" style=\"height:200px;\"\n src=\"https://drive.google.com/uc?export=view&id=
1fSzKGQfWndyXMUUC9zmsDMvqrOHgsf7d\">\n\n <!-- CONVERTIDOR 3,3 V-->\n <h3 class=\"text_limitat\">Convertidor lineal</h3>\n <p class=\"text_limitat\">No n'hi ha prou de tenir 5 V a la sortida d'aquest convertidor. Els 5 V\n són necessaris per alimentar l'Arduino Nano, ja que el seu microcontrolador ATMEGA328 funciona\n a 5 V. La petita placa que conté l'ESP8266 i la placa de l'ADS8232\n van alimentades a 3,3 V. Per tant, és necessari disposar d'una tensió d'alimentació de 3,3 V.\n </p>\n\n <img class=\"text_limitat\" style=\"height:200px;\"\n src=\"https://drive.google.com/uc?export=view&id=
1AuJwkozRa4aLbtxMsWHGX_w2L3Sn2Uql\">\n\n <p class=\"text_limitat\">\n S'opta per un regulador lineal de bona eficiència anomenat AMS1117-3.3. Està especialment pensat\n per passar de tensions de 5 V a 3,3 V, que és exactament el que necessitem. A la seva entrada \n hi connectarem un condensador de 10 uF, així com a la seva sortida. \n Aquests condensadors evitaran la caiguda de tensió a la sortida en pics d'intensitat a la sortida.\n </p>\n\n\n <!-- AD8232-->\n <h3 class=\"text_limitat\">AD8232</h3>\n <p class=\"text_limitat\">La part d'instrumentació de la placa dona una senyal analògica a partir\n de la senyal captada als tres elèctrodes. \n Per això es fa servir una placa de desenvolupament que conté un integrat anomenat AD8232.\n </p>\n\n <img class=\"text_limitat\" style=\"height:200px;\"\n src=\"https://drive.google.com/uc?export=view&id=
1EaA605i9rI-t80x6dUTh64XHpr6J7Ve-\">\n\n <p class=\"text_limitat\">En essència, el que es fa és alimentar la placa i llegir les \n tensions als tres elèctrodes. Dues d'aquestes tensions es resten i el \n resultat s'amplifica amb un amplificador diferencial, la referència del qual és el tercer elèctrode.\n </p>\n\n <!-- ESP-01-->\n <h3

```
    class=\"text_limitat\">ESP-01</h3>\n <p
    class=\"text_limitat\">L'integrat ESP8266 muntat en una placa amb
    antena, cristall oscillador, \n resistències de pullup i
    condensadors de desacoblament es coneix com a ESP-01 i el seu
    pinout.\n </p>\n\n <img class=\"text_limitat\"
    style=\"height:200px;\"\n
    src=\"https://drive.google.com/uc?export=view&id=
1nRSj8TfxQX8b6YnCC7QlvUsoBs0WtlVC\">\n\n <p
    class=\"text_limitat\">El fet de què aquesta placa només tingui 8
    pins no vol dir que el \n component només tingui 8 pins, de fet
    en té al voltant de 30. La placa que utilitzem té \n dos pins
    d'alimentació, un pin de TX i un de \n RX, dos pins de caràcter
    general, un pin per fer reset i un pin per habilitar
    l'integrat.\n </p>\n\n <!--Arduino NANO-->\n <h3
    class=\"text_limitat\">Arduino Nano</h3>\n <p
    class=\"text_limitat\">L'integrat ESP8266 muntat en una placa amb
    antena, cristall oscillador, \n resistències de pullup i
    condensadors de desacoblament es coneix com a ESP-01 i el seu
    pinout.\n </p>\n\n <img class=\"text_limitat\"
    style=\"height:200px;\"\n
    src=\"https://drive.google.com/uc?export=view&id=
1gQma7MnP8vwtP7156rMhl5s3wbR8Psqq\">\n\n <p
class=\"text_limitat\">El fet de què aquesta placa només tingui 8
    pins no vol dir que el \n component només tingui 8 pins, de fet
    en té al voltant de 30. La placa que utilitzem té \n dos pins
    d'alimentació, un pin de TX i un de \n RX, dos pins de caràcter
    general, un pin per fer reset i un pin per habilitar
    l'integrat.\n </p>\n\n <!-- Anell de LEDs -->\n <h3
    class=\"text_limitat\">Anell de LEDs</h3>\n <p
    class=\"text_limitat\">S'ha previst disposar d'un anell de LEDs
    per visualitzar de forma \n ràpida i intuïtiva la freqüència
    cardíaca de la persona que porti el dispositiu. Per bé que no
    podrà\n conèixer amb excessiva precisió la freqüència cardíaca de
    la persona, se'n pot fer una bona idea.\n
</p>\n\n  <img class=\"text_limitat\" style=\"height:200px;\"\n
    src=\"https://drive.google.com/uc?export=view&id=
1dXnm-I8f1N_SSJA29Ub9T15MHDirklRu\">\n\n <p
    class=\"text_limitat\">L'anell que s'ha escollit és de la casa
    SparkFun. Té 16 LEDs 5050 disposats\n de forma circular. S'ha
    d'alimentar a 5 V i mitjançant un pin de DATA\\_IN es pot fer
    el\n control dels colors de tots els LEDs així com de la seva
    intensitat llumínica i la seva saturació.\n </p>\n\n\n\n <p
    class=\"text_limitat\"> </br></p> </div> \n \n <div id=\"teoria\"
    class=\"teoria\">\n <h2
    class=\"text_limitat\">Electrocardiograma</h2>\n <p
    class=\"text_limitat\">De forma teòrica, un electrocardiograma és
    una
mesura indirecta de \n l'activitat elèctrica cardíaca. De fet, és la
    única mesura no invasiva de què es disposa per aquest fi. \n
    Permet identificar alteracions anatòmiques, el ritme, alteracions
    iòniques...\n </p>\n <p class=\"text_limitat\">Durant la
    despolarització del miòcit cardíac es genera una diferència \n de
    potencial de 90 mV. El camp elèctric que es genera és captat pels
    elèctrodes. Aquesta senyal \n elèctrica s'amplifica per tal
    d'aprofitar el rang \n dels convertidors analògics digitals de
    què disposen els electrocardiògrafs digitals.\n </p>\n\n <img
    class=\"text_limitat\" style=\"height:200px;\"\n
    src=\"https://drive.google.com/uc?export=view&id=
```

```
      1EOcIxBEm0G_6diMqYsgq4QYmasYX7arc\">\n\n <p
         class=\"text_limitat\">Hi ha molta literatura que permet
         identificar malalties analitzant \n els diferents segments, la
         seva durada i l'amplitud de la senyal. Tot i que considerem que
         \n es podria programar un algorisme per fer un anàlisi de la
         senyal, \n ens hem centrat en calcular la freqüència cardíaca en
         batecs per minut.\n </p>\n <img class=\"text_limitat\"
         style=\"height:200px;\"\n src=\"https://drive.google.com/uc?
      export=view&id=
      14taRHhw0VruvAXfy3oI7wcPZ6cPwhSoV\">\n\n <p
         class=\"text_limitat\">Per calcular la freqüència cardíaca es
         defineixen dos nivells d'amplitud\n de la senyal analògica
         captada. Quan se supera el llindar superior vol dir que hi ha
         hagut un nou\n batec i s'indica canviant l'estat d'un booleà. A
         més, es calcula el temps respecte el batec anterior.\n Quan es
         baixa d'el llindar inferior es canvia el valor del booleà,
         indicant que ja s'ha donat el \n pic del batec; ens preparem pel
         següent batec.\n </p>\n <p class=\"text_limitat\"> </br></p>
         </div>\n\n <div class=\"footer\">\n <span style=\"float:left;
         margin-top: 10 px; margin-left:
         10px;\">Electrocardiograma</span>\n <span style=\"float:right;
         margin-right: 25px;\">Llorenç Fanals Batllori - Pol Fernández
         Rejón</span>\n <div class=\"s-m\">\n <a class=\"fab fa-github\"
         href=\"https://github.com/LFanals/ECG_WiFi\"></a>\n\n </div>\n\n
         </div>\n\n\n </body>\n</html>\n\n");
   //Serial.println(millis());

         break; // Sortim del if (LiniaActual.length() == 0)
        }
        else { // si tens una nova línia, neteja LiniaActual
         LiniaActual = "";
        }
      }

      else if (c != '\r') { // Si tens algun caràcter afegiex-lo al final
         de LiniaActual
       LiniaActual += c;
      }
     }
   }

   // Tanquem la connexió, esperant un nou client o que el client existent
      refresqui la pàgina
   client.stop();
   Serial.println("Client desconnectat.");
   Serial.println("");
  }
  // Mirem si cal actualitzar els minuts i les hores i si cal fer una
     lectura de tensions
  comprova_temps();

}

void comprova_temps() {
  if ((millis() - millis_anteriors) >= 60000) { // ha passat un minut
  // minuts_actual = (millis() - millis_anteriors) / 60000; //
     minuts_actual que sigui float i que guardi segons
   minuts_actual++;
```

```
    // Serial.print("  MINUTS"); Serial.println(minuts_actual);
    millis_anteriors = millis(); // memoritzem el moment en què això ha
        passat
    if (minuts_actual >= 60) { // si portem 60 minuts, diem que en portem 0
        i incrementem l'hora
      minuts_actual = 0;
      hora_actual++;
      sumatori_2 = 0; n_dades_hora = 0;
      //    lectura_tensions(); // cridem la funció que llegeix les tensions
    }
    if (hora_actual >= 24) { // si l'hora és 24, la passem a 0
      hora_actual = 0;
    }
  }
}
```